UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/815,478	03/31/2004	James Loran Ball	ALTRP134/A1466	6370

51501 7590 05/29/2007

BEYER WEAVER LLP
ATTN: ALTERA
P.O. BOX 70250
OAKLAND, CA 94612-0250

EXAMINER	
LAI, VINCENT	

ART UNIT	PAPER NUMBER
2181	

MAIL DATE	DELIVERY MODE
05/29/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)
	10/815,478	BALL, JAMES LORAN
	Examiner	Art Unit
	Vincent Lai	2181

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *05 March 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08)
Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Continued Examination Under 37 CFR 1.114

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 5 March 2007 has been entered.

Response to Amendment

2. Acknowledgment is made of the amendment to title and claims. All objections are withdrawn after considering amendments.

Response to Arguments

3. Applicant's arguments filed 5 March 2007 have been fully considered but they are not entirely persuasive.

The 35 USC 112 rejections is withdrawn after Applicant has stated an approximate amount of instructions meant to be encompassed by the term "substantially all."

The arguments pertaining to the new added amendments are moot in view of the new rejection presented below.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-11, and 13-29 rejected under 35 U.S.C. 103(a) as being unpatentable over Intel, Inc (IA-32® Architecture Software Developer's Manual, Volumes 1-2, 2002), herein referred to as Intel, in view of Killian et al (U.S Patent # 5,420,992), herein referred to as Killian.

Regarding **independent claim 1**, Intel teaches *a processor, comprising: a plurality of registers* [see Intel, Vol. 1, Page 3-8, section 3.4]*; circuitry configured to process a plurality of instructions* [see Intel, Vol. 1, Page 2-14, Section 2.6.2] *associated with an instruction set including a plurality of branch and non-branch instructions* [see Intel, Vol. 2, section 3.2, starting on page 3-15; Examiner's note: section 3.2 provides a listing of all instruction able to be processed by the P6 architecture, including branch (i.e., JMP, Jcc, CALL, et al.) and non-branch instructions (i.e., ADD, AND, CMP, et al.).], *the plurality of instructions each having a multi-byte length* [see Intel, Vol. 2, page 2-1,

section 2.1], *the plurality of instructions accessible at multi-byte aligned addresses* [see Intel, Vol. 1, Page 1-7, Fig. 1-1; Examiner's note: Since the IA-32 architecture employs 32-bit instructions, these instructions would be accessed by multi-byte aligned addresses.]; *wherein substantially all multi-byte aligned branch instructions are operable to access the instructions at byte aligned addresses* [see Intel, Vol. 2, page 3-357 "JMP-Jump" instruction reference; page 3-358, line 1-2, "A relative offset (rel8, rel16, or rel32) is generally specified as a label in assembly code, but at the machine code level, it is encoded as a signed 8-, 16-, or 32-bit immediate value."; Examiner's note: In the description of operating modes, Intel discloses a jump instruction that uses an offset corresponding to 8 bits (JMP rel8) as well as other indexing modes (rel16, rel32 et al.)] .

Intel does note teach wherein the circuitry is operable perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions.

Killian teaches an ALU that is configured to perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions (See column 8, lines 7-13: A branch that contains an immediate field is sign extended view an ALU immediate computation).

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Intel to include an ALU that is configured to perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for

branch instructions. Intel already teaches the necessity of sign extending branch immediates (See Intel, Vol. 2,page 3-358, line 1-2: "A relative offset (rel8, rel16, or rel32) is generally specified as a label in assembly code, but at the machine code level, it is encoded as a signed 8-, 16-, or 32-bit immediate value."). Sign extensions of immediates are done for various types of operations within the ALU in the Intel architecture. One having ordinary skill in the art would recognize that having an ALU perform sign extensions of branches would be very useful in not having to duplicate hardware in order to perform such operations.

It is noted that Applicant appears to allude to the fact that although having circuitry that is operable perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions is not readily done due to costs, it has been a consideration and thus would not be entirely novel.

Regarding **claim 2**, Intel discloses *the processor of claim 1, wherein the plurality of instructions are accessed at word aligned addresses* [see Intel, Vol. 2, Page 3-358, line 1-2, "...it is encoded as a signed 8-, 16-, or 32-bit immediate value."; Examiner's note: Intel discloses a 32-bit offset, thus word aligned addresses.].

Regarding **claim 3**, Intel discloses *a processor of claim 1, wherein the plurality of instructions are accessed at half-word aligned addresses* [see Intel, Vol. 2, Page 3-358,

line 1-2, "...it is encoded as a signed 8-, 16-, or 32-bit immediate value."; Examiner's note: Intel discloses a 16-bit offset, thus half-word aligned addresses.].

Regarding **claim 4**, Intel discloses *the processor of claim 1, wherein accessing the instructions comprises reading and writing the addresses* [see Intel, Vol. 2, Page 3-357; lines 1-7, "Transfers program control to...a memory location"; Vol. 2, Page 3-359, Operation Code, line 4, "tempEIP <- EIP + DEST"; Examiner's note: In the operation of the jump instruction, Intel discloses reading the address (offset or absolute) from the instruction, as illustrated by "DEST", and writing the address to "tempEIP" for use in changing the instruction pointer.].

Regarding **claim 5**, Intel discloses *the processor of claim 1, wherein branch instructions comprise branch and conditional branch instructions* [see Intel, Vol. 2, section 3.2, instructions (sections) Jcc (conditional jump) and JMP (jump)].

Regarding **claim 6**, Intel discloses *the processor of claim 1, wherein branch instructions comprise a branch offset and a current program counter value* [see Intel, Vol. 2, Page 3-359, Operation Code, line 4, "tempEIP <- EIP + DEST"; Examiner's note: In this cite, Intel discloses an offset (DEST) being added to the program counter value (EIP).].

Regarding **claim 7**, Intel discloses *the processor of claim 1, wherein the units of the branch offset* [see Intel, Vol. 2, Page 3-357, "JMP rel8", "When executing a near jump the processor jumps to the address...that is specified with the target operand"] *and the current program counter are in bytes* [see Intel, Vol. 1, Page 3-8, section 3.4, lines 9-10, "EIP (instruction pointer) register...contains a 32-bit pointer..."; Examiner's note: A 32-bit value is comprised of four 8-bit bytes.].

Regarding **claim 8**, Intel discloses *the processor of claim 1, wherein the plurality of instructions are one word in length* [see Intel, Vol. 1, Page 1-7, Fig. 1-1; Examiner's note: It would have been well known that the IA-32 architecture utilizes 32-bit instructions.].

Regarding **claim 9**, Intel discloses *the processor of claim 1, wherein the branch instruction and a plurality of non-branch instructions supported by the processor are implemented using common subcircuitry* [see Intel, Vol. 1, Page 2-10, Figure 2-1, element "Execution Out-of-Order Core"; Examiner's note: More details of the inner workings of the P6 execution unit are disclosed in the "P6 Family of Processors Hardware Developers Manual" also by Intel, Inc. September 1998, order number 244001-001 (Page 2-5)].

Regarding **claim 10**, Intel discloses *the processor of claim 9, wherein common subcircuitry* [see Intel, Vol. 1, Page 2-10, Figure 2-1, element "Execution Out-of-Order

Core"; Vol. 1, Page 2-14, section 2.6.2;] *is used to handle the immediate field associated with the branch and non-branch instructions* [see Intel, Vol. 2, page 3-21, lines 2-3 "The destination operand can be a register or a memory location; the source operand can be an immediate, a register, or a memory location." (use of immediate processing with non-branch (ADD) instructions); Page 3-357, lines 3-4 "This operand can be an immediate value, a general-purpose register, or a memory location." (use of immediate processing with branch instructions. Examiner's note: It is clear from the Intel disclosure and would have been well known at the time of invention that the P6 processor employs sub circuitry (the execution core) to perform multiple operations, including branch and non-branch instructions. Furthermore, since the IA-32 architecture utilizes immediate fields in both branch and non-branch (i.e., adding an immediate value) instructions, said instructions would both be executed by said sub circuitry, such as an adder to compute the addition or target address, as was common knowledge at the time of invention.].

Regarding **claim 11**, Intel discloses *the processor of claim 10, wherein common subcircuitry is used to perform sign-extensions of the immediate field associated with the branch and non-branch instructions* [see Intel, Vol. 2, Page 3-3, point 4, "**imm8**—An immediate byte value. The **imm8** symbol is a signed number between –128 and +127 inclusive. For instructions in which **imm8** is combined with a word or doubleword operand, the immediate value is sign-extended to form a word or doubleword. The upper byte of the word is filled with the topmost bit of the immediate value." Examiner's

note: As cited multiple time in this action, many instructions (branch and non-branch) utilize an immediate byte value thus would be sign extended by the execution core.].

Regarding **claim 13**, Intel discloses *the processor of claim 1, wherein the processor is a processor core on a* [sic] *ASIC* [Examiner's note: The P6 chip is considered an ASIC, and therefore anticipates the claim.].

Regarding **independent claim 14**, Intel teaches *a processor, comprising: a plurality of registers* [see Intel, Vol. 1, Page 3-8, section 3.4]*; circuitry* [see Intel, Vol. 1, Page 2-14, Section 2.6.2] *configured to process a plurality of branch and non-branch instructions associated with an instruction set* [see Intel, Vol. 2, section 3.2, starting on page 3-15; Examiner's note: section 3.2 provides a listing of all instruction able to be processed by the P6 architecture, including branch (i.e., JMP, Jcc, CALL, et al.) and non-branch instructions (i.e., ADD, AND, CMP, et al.).], *the plurality of branch instructions and non-branch instructions including an immediate field* [see Intel, Vol. 2, Page 3-21, line "Add *imm8* to AL"; Page 3-357, lines 3-4 "This operand can be an immediate value, a general-purpose register, or a memory location."]*; wherein common subcircuitry* [see Intel, Vol. 1, Page 2-10, Figure 2-1, element "Execution Out-of-Order Core"; Vol. 1, Page 2-14, section 2.6.2;] *is used to process the immediate field associated with one or more branch instructions and one or more non-branch instructions* [see Intel, Vol. 2, page 3-21, lines 2-3 "The destination operand can be a register or a memory location; the source operand can be an immediate, a register, or a

memory location." (use of immediate processing with non-branch (ADD) instructions); Page 3-357, lines 3-4 "This operand can be an immediate value, a general-purpose register, or a memory location." (use of immediate processing with branch instructions). Examiner's note: It is clear from the Intel disclosure and would have been well known at the time of invention that the P6 processor employs sub circuitry (the execution core) to perform multiple operations, including branch and non-branch instructions. Furthermore, since the IA-32 architecture utilizes immediate fields in both branch and non-branch (i.e., adding an immediate value) instructions, said instructions would both be executed by said sub circuitry, such as an adder to compute the addition or target address, as was common knowledge at the time of invention.].

Intel does note teach wherein the circuitry is operable perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions.

Killian teaches an ALU that is configured to perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions (See column 8, lines 7-13: A branch that contains an immediate field is sign extended view an ALU immediate computation).

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Intel to include an ALU that is configured to perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for

branch instructions. Intel already teaches the necessity of sign extending branch immediates (See Intel, Vol. 2,page 3-358, line 1-2: "A relative offset (rel8, rel16, or rel32) is generally specified as a label in assembly code, but at the machine code level, it is encoded as a signed 8-, 16-, or 32-bit immediate value."). Sign extensions of immediates are done for various types of operations within the ALU in the Intel architecture. One having ordinary skill in the art would recognize that having an ALU perform sign extensions of branches would be very useful in not having to duplicate hardware in order to perform such operations.

It is noted that Applicant appears to allude to the fact that although having circuitry that is operable perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions is not readily done due to costs, it has been a consideration and thus would not be entirely novel.

Regarding **claim 15**, Intel discloses *the processor of claim 14, wherein the instruction set comprises a plurality of instructions* [see Intel, Vol. 2, section 3.2 (listing of a plurality of instructions supported by the P6 architecture.].

Regarding **claim 16**, Intel discloses *the processor of claim 15, wherein the plurality of instructions are accessed at half-word aligned addresses* [see Intel, Vol. 2, Page 3-358, line 1-2, "...it is encoded as a signed 8-, 16-, or 32-bit immediate value"; Examiner's note: Intel discloses a 16-bit offset, thus half-word aligned addresses.].

Regarding **claim 17**, Intel discloses *the processor of claim 14, wherein branch instructions comprise branch and conditional branch instructions* [see Intel, Vol. 2, section 3.2, instructions (sections) Jcc (conditional jump) and JMP (jump)].

Regarding **claim 18**, Intel discloses *the processor of claim 14, wherein common subcircuitry* [see Intel, Vol. 1, Page 2-10, Figure 2-1, element "Execution Out-of-Order Core"; Vol. 1, Page 2-14, section 2.6.2;] *is used to handle the immediate field associated with the branch and non-branch instructions* [see Intel, Vol. 2, page 3-21, lines 2-3 "The destination operand can be a register or a memory location; the source operand can be an immediate, a register, or a memory location." (use of immediate processing with non-branch (ADD) instructions); Page 3-357, lines 3-4 "This operand can be an immediate value, a general-purpose register, or a memory location." (use of immediate processing with branch instructions). Examiner's note: It is clear from the Intel disclosure and would have been well known at the time of invention that the P6 processor employs sub circuitry (the execution core) to perform multiple operations, including branch and non-branch instructions. Furthermore, since the IA-32 architecture utilizes immediate fields in both branch and non-branch (i.e., adding an immediate value) instructions, said instructions would both be executed by said sub circuitry, such as an adder to compute the addition or target address, as was common knowledge at the time of invention.].

Regarding **claim 19**, Intel discloses *the processor of claim 18, wherein common subcircuitry is used to perform sign-extensions of the immediate field associated with the branch and non-branch instructions* [see Intel, Vol. 2, Page 3-3, point 4, "**imm8**—An immediate byte value. The **imm8** symbol is a signed number between –128 and +127 inclusive. For instructions in which **imm8** is combined with a word or doubleword operand, the immediate value is sign-extended to form a word or doubleword. The upper byte of the word is filled with the topmost bit of the immediate value." Examiner's note: As cited multiple time in this action, many instructions (branch and non-branch) utilize an immediate byte value thus would be sign extended by the execution core.].

Regarding **independent claim 20**, Intel teaches *a method for performing an instruction, the method comprising: decoding a branch instruction associated with an address* [see Intel, Vol. 1, Page 2-10, Fig. 2-1, element "Fetch/Decode"], *the branch instruction having an associated opcode and an immediate value* [see Intel, Vol. 2, Page 3-357, heading of table, "Opcode Instruction Description", Page 3-357, lines 3-4, "This operand can be an immediate value..."]; *calculating a branch target address using the immediate value* [see Intel, Vol. 2, Page 3-359, Operation Code, line 4, "tempEIP <- EIP + DEST"; Examiner's note: In this cite, Intel discloses an offset (DEST) being added to the program counter value (EIP).], *wherein the branch target address is determined by using common subcircuitry, the common subcircuitry operable to calculate a byte-aligned address* [see Intel, Vol. 2, Page 3-359, Operation Code, line 4, "tempEIP <- EIP + DEST"; Examiner's note: In this cite, Intel discloses an offset (DEST) being added to

the program counter value (EIP); Examiner's note: It is clear that since Intel allows for a byte to be used as the offset.], *wherein the common subcircuitry is also configured to perform nonbranch operations* [see Intel, Vol. 1, Page 2-10, Figure 2-1, element "Execution Out-of-Order Core"; Examiner's note: More details of the inner workings of the P6 execution unit are disclosed in the "P6 Family of Processors Hardware Developers Manual" also by Intel, Inc. September 1998, order number 244001-001 (Page 2-5). Furthermore, since Intel discloses multiple instructions (branch and non-branch, see Vol. 2, section 3.2), it is clear that both of these types of instructions are executed by the execution core.]; *jumping to the branch target address, wherein the branch target address is multi-byte aligned* [see Intel, Vol. 2, Page 3-357, lines 1-2; Examiner's note: In the case of a jump instruction with a 32 bit immediate, a branch target would be fetched that is multi-byte aligned.].

Intel does note teach wherein the circuitry is operable perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions.

Killian teaches an ALU that is configured to perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions (See column 8, lines 7-13: A branch that contains an immediate field is sign extended view an ALU immediate computation).

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Intel to include an ALU that is configured to

perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions. Intel already teaches the necessity of sign extending branch immediates (See Intel, Vol. 2,page 3-358, line 1-2: "A relative offset (rel8, rel16, or rel32) is generally specified as a label in assembly code, but at the machine code level, it is encoded as a signed 8-, 16-, or 32-bit immediate value."). Sign extensions of immediates are done for various types of operations within the ALU in the Intel architecture. One having ordinary skill in the art would recognize that having an ALU perform sign extensions of branches would be very useful in not having to duplicate hardware in order to perform such operations.

It is noted that Applicant appears to allude to the fact that although having circuitry that is operable perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions is not readily done due to costs, it has been a consideration and thus would not be entirely novel.

Regarding **claim 21**, Intel discloses *the method of claim 20, wherein the branch target address is multi-byte aligned* [see Intel, Vol. 2, Page 3-357, lines 1-2; Examiner's note: In the case of a jump instruction with a 32 bit immediate, a branch target would be fetched that is multi-byte aligned.].

Regarding **claim 22**, Intel discloses *the method of claim 20, wherein the branch target address is half-word aligned* [see Intel, Vol. 2, Page 3-358, line 1-2, "...it is encoded as a signed 8-, 16-, or 32-bit immediate value."; Examiner's note: Intel discloses a 16-bit offset, thus half-word aligned addresses.].

Regarding **claim 23**, Intel discloses *the method of claim 20, wherein calculating the branch target address comprises performing a sign extend operation* [see Intel, Vol. 2, Page 3-3, point 4, "**imm8**—An immediate byte value. The **imm8** symbol is a signed number between –128 and +127 inclusive. For instructions in which **imm8** is combined with a word or doubleword operand, the immediate value is sign-extended to form a word or doubleword. The upper byte of the word is filled with the topmost bit of the immediate value." Examiner's note: As cited multiple time in this action, many instructions (branch and non-branch) utilize an immediate byte value thus would be sign extended by the execution core.].

Regarding **claim 24**, Intel discloses *the method of claim 20, wherein the branch instruction calculates the branch target address using the immediate value and the address of the branch instruction* [see Intel, Vol. 2, Page 3-357; lines 1-7, "Transfers program control to...a memory location"; Vol. 2, Page 3-359, Operation Code, line 4, "tempEIP <- EIP + DEST"; Examiner's note: In the operation of the jump instruction, Intel discloses reading the address (offset or absolute) from the instruction, as illustrated

by "DEST", and writing the address to "tempEIP" for use in changing the instruction pointer.].

Regarding **claim 25**, Intel discloses *the method of claim 20, wherein the units of the immediate value* [see Intel, Vol. 2, Page 3-357, "JMP rel8", "When executing a near jump the processor jumps to the address...that is specified with the target operand"] *and the address associated with the branch instruction are in bytes* [see Intel, Vol. 1, Page 3-8, section 3.4, lines 9-10, "EIP (instruction pointer) register...contains a 32-bit pointer..."; Examiner's note: A 32-bit value is comprised of four 8-bit bytes.]..

Regarding **claim 26**, Intel discloses *the method of claim 25, wherein the address associated with the branch instruction is a program counter* [see Intel, Vol. 1, Page 3-8, section 3.4, lines 9-10, "EIP (instruction pointer) register...contains a 32-bit pointer..."].

Regarding **independent claim 27**, Intel teaches *a processor, comprising: means for decoding* [see Intel, Vol. 1, Page 2-10, Fig. 2-1, element "Fetch/Decode"] *a branch instruction associated with an address* [see Intel, Vol. 2, Page 3-357; Examiner's note: Intel discloses one of a plurality of types of branch instructions in this instruction definition.], *the branch instruction having an associated opcode and an immediate value* [see Intel, Vol. 2, Page 3-357, heading of table, "Opcode Instruction Description", Page 3-357, lines 3-4, "This operand can be an immediate value..."]; *means for calculating a branch target address using the immediate value* [see Intel, Vol. 2, Page 3-359,

Operation Code, line 4, "tempEIP <- EIP + DEST"; Examiner's note: In this cite, Intel discloses an offset (DEST) being added to the program counter value (EIP).], *wherein the branch target address is determined by using common subcircuitry, the common subcircuitry operable to calculate a byte-aligned address* [see Intel, Vol. 2, Page 3-359, Operation Code, line 4, "tempEIP <- EIP + DEST"; Examiner's note: In this cite, Intel discloses an offset (DEST) being added to the program counter value (EIP); Examiner's note: It is clear that since Intel allows for a byte to be used as the offset.], *wherein the common subcircuitry is also configured to perform nonbranch operations* [see Intel, Vol. 1, Page 2-10, Figure 2-1, element "Execution Out-of-Order Core"; Examiner's note: More details of the inner workings of the P6 execution unit are disclosed in the "P6 Family of Processors Hardware Developers Manual" also by Intel, Inc. September 1998, order number 244001-001 (Page 2-5). Furthermore, since Intel discloses multiple instructions (branch and non-branch, see Vol. 2, section 3.2), it is clear that both of these types of instructions are executed by the execution core.]; *means for jumping to the branch target address, wherein the branch target address is multi-byte aligned* [see Intel, Vol. 2, Page 3-357, lines 1-2; Examiner's note: In the case of a jump instruction with a 32 bit immediate, a branch target would be fetched that is multi-byte aligned.].

Intel does note teach wherein the circuitry is operable perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions.

Killian teaches an ALU that is configured to perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in

branch instructions to calculate a target address for branch instructions (See column 8, lines 7-13: A branch that contains an immediate field is sign extended view an ALU immediate computation).

It would have been obvious to a person having ordinary skill in the art at the time the invention was made to have modified Intel to include an ALU that is configured to perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions. Intel already teaches the necessity of sign extending branch immediates (See Intel, Vol. 2,page 3-358, line 1-2: "A relative offset (rel8, rel16, or rel32) is generally specified as a label in assembly code, but at the machine code level, it is encoded as a signed 8-, 16-, or 32-bit immediate value."). Sign extensions of immediates are done for various types of operations within the ALU in the Intel architecture. One having ordinary skill in the art would recognize that having an ALU perform sign extensions of branches would be very useful in not having to duplicate hardware in order to perform such operations.

It is noted that Applicant appears to allude to the fact that although having circuitry that is operable perform sign extensions of immediate fields in non-branch instructions and perform sign extensions of immediate fields in branch instructions to calculate a target address for branch instructions is not readily done due to costs, it has been a consideration and thus would not be entirely novel.

Regarding **claim 28**, Intel discloses *the processor of claim 27, wherein the branch target address is multi-byte aligned* [see Intel, Vol. 2, Page 3-357, lines 1-2; Examiner's note: In the case of a jump instruction with a 32 bit immediate, a branch target would be fetched that is multi-byte aligned.].

Regarding **claim 29**, Intel discloses *the processor of claim 27, wherein the branch target address is half-word aligned* [see Intel, Vol. 2, Page 3-358, line 1-2, "...it is encoded as a signed 8-, 16-, or 32-bit immediate value."; Examiner's note: Intel discloses a 16-bit offset, thus half-word aligned addresses.].

5. Claims 12 and 30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Intel in view of Killian in view of Solomon et al (US Pat. No. 6,219,833; herein referred to as "Solomon").

Regarding **claim 12**, Intel discloses the limitations as stated in **independent claim 1**.

Intel does not disclose *the processor* [being] *a processor core on a programmable chip.*

Solomon does disclose *the processor* [being] *a processor core on a programmable chip* [see Solomon, Col. 4, lines 46-50; lines 63-66].

The advantage of utilizing a processor core as that disclosed by Intel in the environment of a programmable chip would have been to utilize the general purpose

nature of a chip such as that as the chip executing IA-32 instructions. Furthermore, the use of a programmable core in conjunction with a fixed processing core would have allowed one to develop a system capable of performing specific functions faster (such as DSP algorithms). Solomon discloses the use of an Intel Pentium II processor as the primary fixed processor, therefore it would have been obvious to one of ordinary skill in the art at the time of invention to utilize the processor disclosed by Intel with a secondary programmable core on the same chip.

Regarding **claim 30**, Intel discloses the limitations as stated in **independent claim 27**.

Intel does not disclose the processor being included in *a programmable chip*.

Solomon does disclose the processor being included in *a programmable chip*.

The advantage of utilizing a processor core as that disclosed by Intel in the environment of a programmable chip would have been to utilize the general purpose nature of a chip such as that as the chip executing IA-32 instructions. Furthermore, the use of a programmable core in conjunction with a fixed processing core would have allowed one to develop a system capable of performing specific functions faster (such as DSP algorithms). Solomon discloses the use of an Intel Pentium II processor as the primary fixed processor, therefore it would have been obvious to one of ordinary skill in the art at the time of invention to utilize the processor disclosed by Intel with a secondary programmable core on the same chip.

Conclusion

6. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Vincent Lai whose telephone number is (571) 272-6749. The examiner can normally be reached on M-F 8:00-5:30 (First BiWeek Friday Off).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Donald Sparks can be reached on (571) 272-4201. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Vincent Lai
Examiner
Art Unit 2181

vl
May 16, 2007

DONALD SPARKS
SUPERVISORY PATENT EXAMINER